Makefile

CMPE 230 - Spring 2024

Gökçe Uludoğan

Example

```c
// main.c
#include <stdio.h>

void manipulate_water();
void manipulate_air();

int main() {
    printf("Welcome to the Avatar-inspired project!\n");
    manipulate_water();
    manipulate_air();
    return 0;
}
```

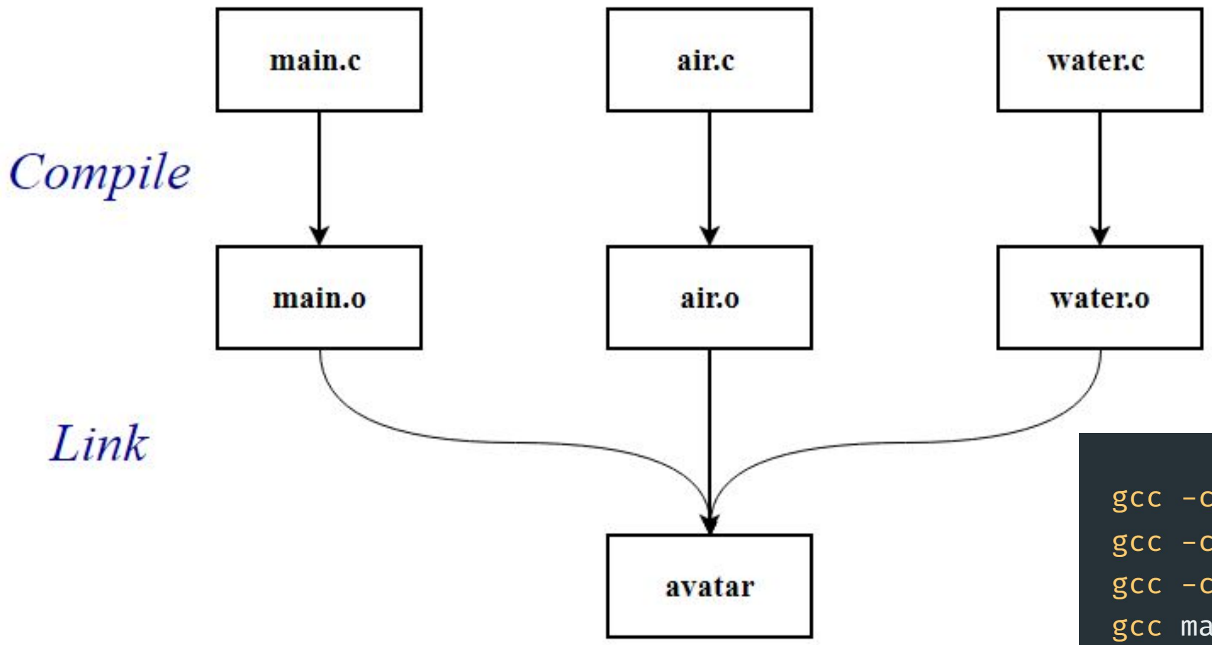
```c
// water.c
#include <stdio.h>

void manipulate_water() {
    printf("Manipulating water...\n");
}
```

```c
// air.c
#include <stdio.h>

void manipulate_air() {
    printf("Manipulating air...\n");
}
```

Compile & Link

```
gcc -c main.c
gcc -c water.c
gcc -c air.c
gcc main.o water.o air.o -o avatar
```

Dependency Diagram



```
gcc -c main.c
gcc -c water.c
gcc -c air.c
gcc main.o water.o air.o -o avatar
```

Makefile

a simple way to organize code compilation

a set of rules to perform

> one or more targets, zero or more dependencies, and zero or more commands in the form:
>
> **target:** dependencies
>
> <tab> commands

make with no arguments executes the first rule in the file

```
avatar:     main.o air.o water.o
            gcc main.o air.o water.o -o avatar

main.o:     main.c
            gcc -c main.c

air.o:      air.c
            gcc -c air.c

water.o:    water.c
            gcc -c water.c
```

Wildcards

```
avatar:     main.o air.o water.o
            gcc main.o air.o water.o -o avatar

main.o:     main.c
            gcc -c main.c

air.o:      air.c
            gcc -c air.c

water.o:    water.c
            gcc -c water.c
```

```
avatar:         main.o air.o water.o
                gcc main.o air.o water.o -o avatar

%.o:            %.c
                gcc -c $*.c
```

Static Linking

The process of linking libraries directly into the executable file.

This creates a **larger executable** file but eliminates the need for the libraries to be present on the system where the executable is run.

```c
#include <stdio.h>
#include <math.h>

void manipulate_water();
void manipulate_air();

int main() {
    double result;
    printf("Welcome to the Avatar-inspired project!\n");
    manipulate_water();
    manipulate_air();
    result = pow(2, 3);
    printf("2^3 = %f\n", result);
    return 0;
}
```

Static Linking

```
avatar:         main.o air.o water.o
                gcc main.o air.o water.o -o avatar -lm

main.o:         main.c
                gcc -c main.c

air.o:          air.c
                gcc -c air.c

water.o:        water.c
                gcc -c water.c
```

<math.h>

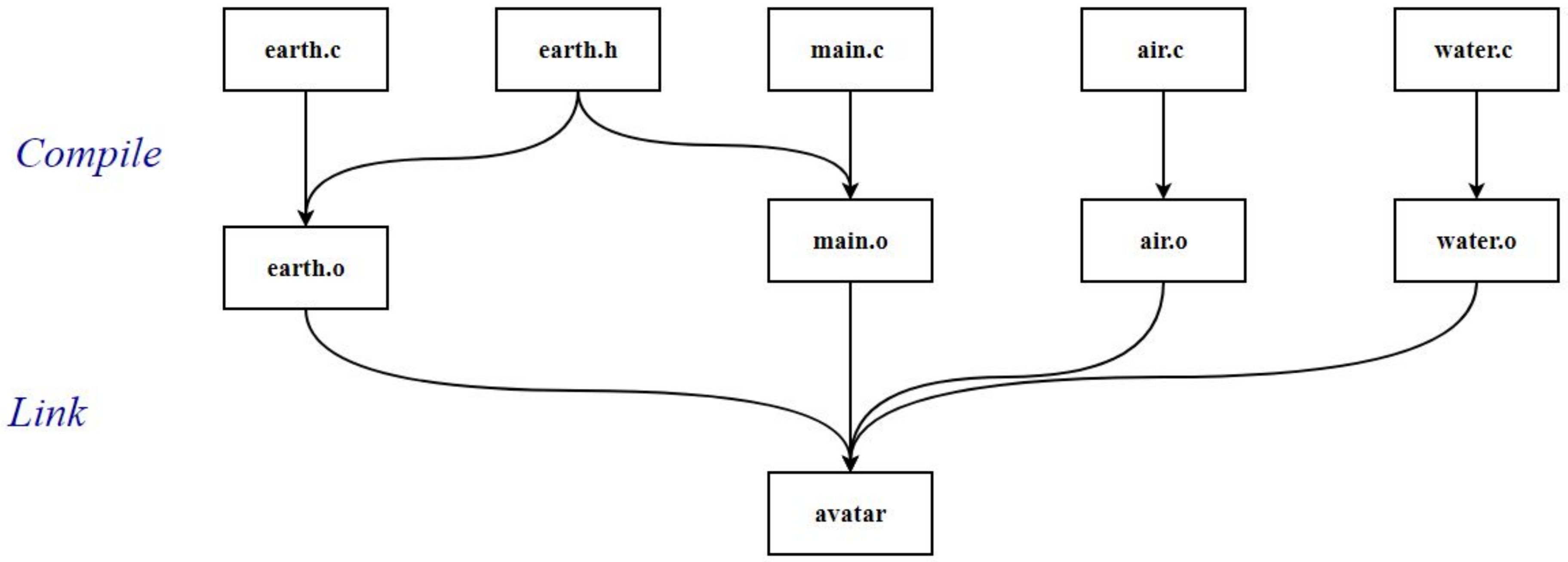
Example 2

```c
// earth.c
#include <stdio.h>
#include "earth.h"

void manipulate_earth() {
    printf("Manipulating earth ... \n");
}
```

```c
#ifndef EARTH_H
#define EARTH_H

void manipulate_earth();

#endif
```

```c
#include <stdio.h>
#include <math.h>
#include "earth.h"

void manipulate_water();
void manipulate_air();

int main() {
    double result;
    printf("Welcome to the Avatar-inspired project!\n");
    manipulate_water();
    manipulate_air();
    manipulate_earth();
    result = pow(2, 3);
    printf("2^3 = %f\n", result);
    return 0;
}
```

Dependency Diagram



Makefile

```
avatar:         main.o air.o water.o earth.o
                gcc main.o air.o water.o earth.o -o avatar -lm

main.o:         main.c earth.h
                gcc -c main.c -I .

air.o:          air.c
                gcc -c air.c

water.o:        water.c
                gcc -c water.c

earth.o:        earth.c earth.h
                gcc -c earth.c -I .
```

Including Files from Different Directories

```
INCDIR=INC

avatar:         main.o air.o water.o earth.o
                gcc main.o air.o water.o earth.o -o avatar -lm

main.o:         main.c $(INCDIR)/earth.h
                gcc -c main.c -I $(INCDIR)

air.o:          air.c
                gcc -c air.c

water.o:        water.c
                gcc -c water.c

earth.o:        $(INCDIR)/earth.c $(INCDIR)/earth.h
                gcc -c $(INCDIR)/earth.c -I $(INCDIR)
```